## REMARKS

This Amendment is filed in connection with a Request for Continued Examination in response to the Decision on Appeal dated Jan. 5, 2010. Pursuant to 37 CFR 1.114 the Applicant requests further examination. New claims are presented herein, that include additional novel limitations not considered by the Board.

Claims 27-46 are now pending in the application.

Claims 1-26 stand cancelled.

New claims 27-46 are presented.

In the Decision on Appeal the Board made the below-reproduced findings of fact as to the content Westberg:

> 2. Westberg relates to improving bandwidth utilization and transmission efficiency associated with the point-to-point transportation of internet protocol (IP) data packets in a network environment by employing asynchronous transfer mode (ATM) adaption layer two (AAL2) minicells as a bearer. (Abstract.)
>
> 3. Westberg discloses a method for transporting an internet protocol data packet over an AAL2 connection wherein the internet protocol (IP) data packet includes a header portion. If the data associated with a session context/connection identifier data field in a header portion of a first IP data packet has not been previously stored in a look-up table, the method inserts the full IP data packet header into a first AAL2 minicell. (Col. 4, ll. 15-26.)

4. Westberg further discloses that, at the receiving point, the data associated with the session context/connection identifier data field is stored in the look-up table in accordance with the unused address in the data field associated with the AAL2. (Col. 4, ll. 30-34.)

5. As depicted in Figure 8, Westberg describes the data fields included in the header 805, such as a session context or connection identifier (ID) 850. (Col. 6, ll. 4-8.)

6. Westberg teaches that, if the source/destination/connection/flow information has not been previously stored in the look-up table, the compression algorithm identifies an unused entry in the look-up table and inserts the address of the unused look-up table entry in the CID data field 304 of the corresponding AAL2 minicell. A full header containing the source/destination/connection/flow information associated with the session context/connection ID 850 is inserted into the payload of the AAL2 minicell and then transferred to the receiving point. (Col. 6, l. 63 – col. 7, l. 8.)

7. Westberg further discloses that the decompression algorithm stores the previously unstored source/destination/connection/flow information in the lookup table based on the address that the compression algorithm stored in the CID field 304. By doing so, the transmitted IP/PPP data packets associated with the same session/connection need only carry the look-up table address in the CID field 304 of the corresponding AAL2 minicell header, rather than the source/destination/connection/flow information. (Col. 7, ll. 10-18.)

The Applicant's new claims include additional novel limitations that are not anticipated or rendered obvious by these findings of fact, or any portion of Westberg, or the other cited references.

For example, the Applicant's new claim 27, recites (emphasis added):

27. (NEW) A method comprising;

executing an application on a network device, the application having a plurality of modules each associated with one or more layers of a hierarchy of communication protocols;

***storing, in a memory space associated with the application, a connection data structure, the connection data structure storing <u>together</u> data for the plurality of modules of the application*** for a connection maintained by the network device;

forming a unique connection identifier for the connection;

***<u>independently checkpointing</u> portions of the connection data structure for different modules into a memory space associated with a checkpoint server, the portions*** of the connection data structure each being embedded with the unique connection identifier and ***<u>stored separately</u> in the memory space associated with the checkpoint server***; and

***in response to a restart or failure of the application executing on the network device, restoring the connection data structure in the memory space associated with the application, by <u>retrieving the separately stored portions</u> of the connection data structure for at least some of the different modules from the memory space associated with a checkpoint server and <u>combining them</u> to reform the connection data structure in the memory space associated with the application.***

The Applicant's claimed technique initially stores, a connection data structure in a memory space associated with an application. This connection data structure stores <u>together</u> data for a plurality of modules of the application. Portions of the connection data structure for different modules are <u>independently checkpointed</u> into a memory space associated with a checkpoint server and <u>stored separately</u> in the memory space associated with the checkpoint server. For example, in reference to the Applicant's Fig 5A (reproduced below) in one embodiment a "Base Layer" portion of the connection data structure, a "TCP" portion of the connection data structure, a "UDP" portion of the connection data structure, etc. may be <u>independently</u> <u>checkpointed</u> into a memory space associated with the checkpoint server and <u>stored separately</u> in the memory space associated with the checkpoint server.

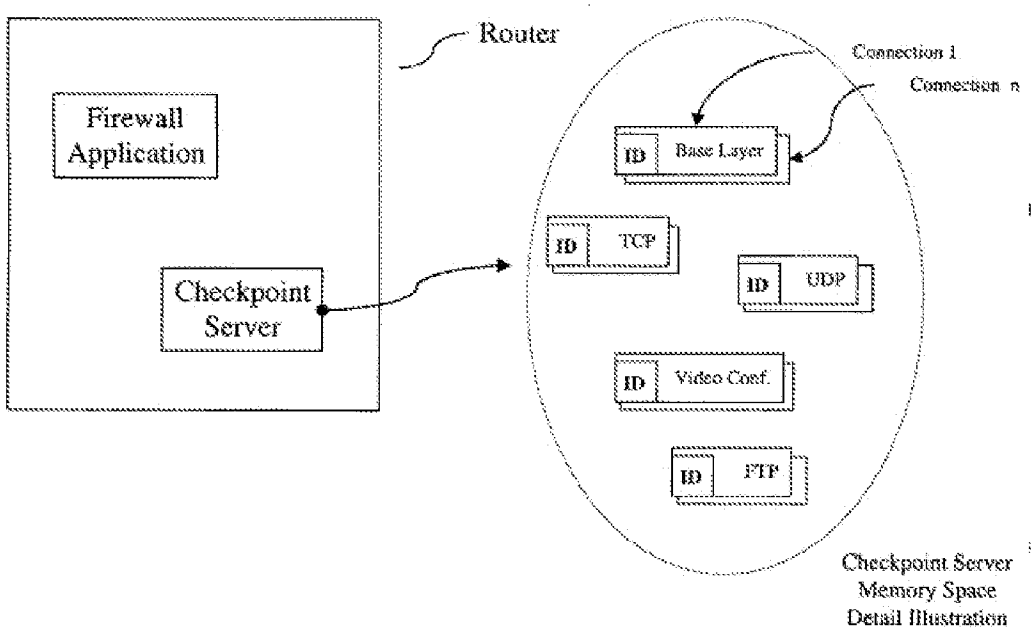


FIG. 5A

In response to a restart or failure of the application executing on the network device (where the connection data structure could be lost), the connection data structure is restored in the memory space associated with the application by retrieving the separately stored portions of the connection data structure for at least some of the different modules from the memory space associated with a checkpoint server and combining them to reform the connection data structure in the memory space associated with the application. For example, in reference to the Applicant's Fig. 5B (reproduced below) in one embodiment a "Base Layer" portion of the connection data structure, a second portion of the connection data structure used by "Module 1", a third portion of the connection data structure used by "Module 2", etc. may be retrieved from separate storage and combined to reform the connection data structure.

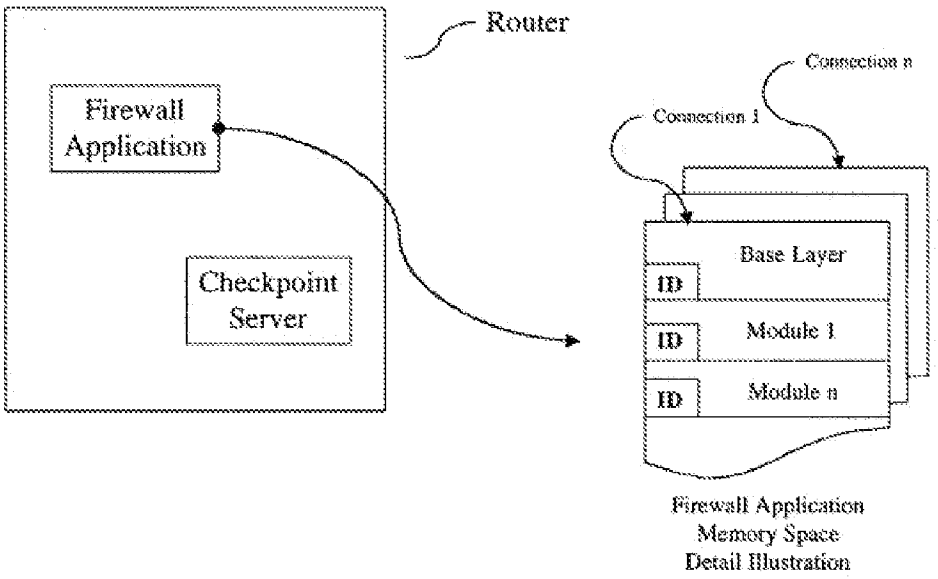


FIG. 5B

Westberg lacks any suggestion of taking data for a plurality of modules of an application originally stored together in a connection data structure in one memory space, and independently checkpointing them to another memory space where the data for the plurality of modules is stored separately. Further, Westberg makes no suggestion of retrieving the separately stored portions of the connection data structure from the other memory space, and combining them to reform the connection data structure in the former memory space after a restart or failure. Indeed, Westberg does not even deal with a checkpointing and restore technique, instead describing a method for transporting data packets where some data from a header portion of a data packet may be stored in a lookup table. This is quite different from the new claims. Accordingly the Applicant respectfully requests favorable action.

In the event that the Examiner deems a telephone conversation desirable in disposition of this application, the Examiner is encouraged to call the undersigned attorney at (617) 951-2500.

In summary, all the independent claims are believed to be in condition for allowance and therefore all dependent claims that depend there from are believed to be in condition for allowance. The Applicant respectfully solicits favorable action.

Please charge any additional fee occasioned by this paper to our Deposit Account No. 03-1237.

Respectfully submitted,

__/James A. Blanchette/_____
James A. Blanchette
Reg. No. 51,477
CESARI AND MCKENNA, LLP
88 Black Falcon Avenue
Boston, MA 02210-2414
(617) 951-2500